EACNET Support for Local Stations

Ethernet console access to token ring front ends

Nov 9, 1991

New accelerator consoles cannot connect to token ring, which is the network of choice for the control system front ends, due to hardware limitations. As a stop-gap measure, so-called DACNET server nodes, which are more expensive Vaxes that *do* connect to token ring, have been used to bridge this gap. There are now commercially available bridges that offer much better performance, and such a bridge will be used instead of DACNET servers. Special accommodations are needed for front ends to support the new EACNET, the term used to refer to ethernet-based Acnet.

Console nodes use ethernet addresses in the form AA-00-04-00-nn-E8. The nn is a node number on "trunk 8". Acnet headers include this source node# as 08nn, where here the trunk is shown in the hi byte. For a request message, the source node is found in the 4th word, whereas for a reply message it is in the 3rd word.

When the CrossComm bridge is used, the ethernet source address is altered by having each byte bit-reversed. So a front end on token ring will see ethernet addresses of the form 55-00-20-00-uu-17, where uu represents the bit-reversed version of nn above. In the future, it may be necessary to use ethernet addresses of a different form, in which uu is *not* the bit-reversed version of nn, or in which the last byte of E8 (or 17) is changed to some other value. The value of the first 4 bytes, however, is not expected to change anytime soon.

To support the required address translation, front ends should keep a table of two-byte values, indexed by nn, which can be used to furnish the last two bytes of a destination ethernet address. The table can be built dynamically, or it can be downloaded from OPER using a suitable ACNAUX protocol variant.

To build this lookup table dynamically, a front end should recognize a frame coming from an ethernet node by its source address field. If it looks like the form 55-00-20-00-uu-xx, and targets a node# of 08nn, it can be assumed that it comes from an ethernet node through the bridge. Record the uu-xx bytes in the table indexed by nn. When there is a reply to a node whose node number is 08nn, do a table lookup to get the two bytes to append to the constant 55-00-20-00 to derive the destination network address. Such an address will pass through the bridge to the corresponding ethernet node.

There is a *caveat* regarding frame size, however. Ethernet frames are limited in length to 1500 bytes. The token ring limit is about 4K bytes. If a frame larger than the ethernet limit is to be sent to a console on ethernet, the front end must send it to a special "packeter" node on token ring, which will in turn forward the it to the ethernet node in packets of less than 1500 bytes. Likewise, when an ethernet

this packeter node, which assembles the packets together and sends the larger frame onto the token ring destination node. In this case, the front end will receive a frame whose source address is that of the packeter node, not the ethernet node.

A large request message might be sent from a console that results in a short reply message. In this case, the ethernet address may not be seen by the front end, so it cannot reply through the bridge. The solution in this case might be to send the reply through the packeter node anyway. Of course, a previous short frame received from that same console may have resulted in placing a proper entry in the table referred to above, so that this case would not happen. Another option would be to form a default ethernet address from the node#.

Local station front ends automatically pack multiple messages that are destined for the same node and do not exceed the frame size limits into common frames, if they are queued to be sent at the same time. The software is organized to build reply frames at the same time so as to increase the likelihood of this occurrence. This is done to reduce the number of frames transmitted, which can pay big dividends for the receiver in terms of reduced network handling soft ware overhead. Vax consoles should expect to receive frames with multiple messages.

This logic of combining multiple messages into common frames brings up another wrinkle in providing support for EACNET in the front ends. Messages destined for node 08nn cannot be combined if some are larger than the ether net limit and some are smaller, since the real destination node is different. This is easily handled by replacing the target node# of large messages with the packeter node#, without altering the destination node# in the acnet header itself.

The table referred to above in OPER can be obtained by sending a request mes sage using the ACNAUX typecode word of 000E. The reply consists of 256 words (512 bytes) indexed by nn, where the 00 entry is unused. At this writing, the table contents are 0000, E801, E802,… , E84F, and the rest are 0000. The table should change only rarely, and it can be downloaded at that time if the front end sup ports the typecode (000F?) used for downloading this table.

Implementation in the local station software

NetLayer module

In the NetQueue routine, the replacement of the target node# with the packeter node# is done for a message larger than 1496 bytes.

NetInt module

manually initialized from scratch, which occurs if the TRING table is destroyed. (Resetting the local station does not alter this non-volatile memory table.) Each table entry includes the six-byte network address and a two-byte diagnostic count of the number of frames sent since the last time a frame was received from that trunk 8 node.

The NetRInt token ring receive interrupt routine only allows source network addresses of the form 40020000xxxx and 55002000xxxx. This insures that a reply to a request can be delivered.

The NetSendF routine prepares the token ring destination address. If the target node# is in the trunk 8 range, the network address is taken from the Node Address Table entry indexed by the node# low byte. In the case that the entry is empty, a default address is built of the form 55002000uu17, where uu is the bit-reversed value of the node# low byte.

ANet module

The Acnet Task analyzes a frame received via SAP $68 and dispatches the acnet header-based messages it contains according to the destination task. If the source network address in the frame header is of the form 55002000uu17, *and* the source node# (found in the acnet header of the first message) is in the trunk 8 range, the six-byte source address is recorded in the Node Address Table entry indexed by the low byte of the source node#, and the entry's count is cleared.

In this first implementation, there is no support for downloading the OPER-based table referred to above, although it could be added in the future if the dynami cally-populating logic proves inadequate.